POPTEX: Interactive Ocean Model Visualization Using Texture Mapping Hardware

Allen M$^{\mathrm{c}}$Pherson
Advanced Computing Laboratory

Mathew Maltrud
Theoretical Division

Los Alamos National Laboratory

Abstract

Global circulation models are used to gain an understanding of the processes that affect the Earth's climate and may ultimately be used to assess the impact of humanity's activities on it. The POP ocean model developed at Los Alamos is an example of such a global circulation model that is being used to investigate the role of the ocean in the climate system. Data output from POP has traditionally been visualized using video technology which precludes rapid modification of visualization parameters and techniques. This paper describes a visualization system that leverages high speed graphics hardware, specifically texture mapping hardware, to accelerate data exploration to interactive rates. We describe the design of the system, the specific hardware features used, and provide examples of its use. The system is capable of viewing ocean circulation simulation results at up to 60 frames per second while loading texture memory at approximately 72 million texels per second.

1 Introduction to POP

The Earth's climate is determined by a complicated interaction between the ocean, sea ice, atmosphere, and biosphere. Computer models that simulate numerically the behavior of this system are one of the best means we have for projecting future climate and the impact of humanity's activities on it. Present-day general circulation models (GCMs) are able to simulate satisfactorily many aspects of the current climate, though a new generation of models is needed that have finer spatial resolution and that more realistically treat the physical processes that control our climate. To meet these objectives, we need GCMs that run on massively parallel computers.

Scientists at Los Alamos have developed one such model: a global ocean circulation model named the Parallel Ocean Program (POP) [3]. POP is based on the widely used Bryan-Cox-Semtner ocean model but was completely rewritten and reformulated for efficient execution on Connection Machine (CM-5) and Silicon Graphics (SGI) Origin 2000 supercomputers.

The global ocean simulation described here employs a grid containing 1280 uniformly spaced points in longitude (0.28° spacing) and 896 variably spaced points from 78°N to 78°S (0.17° average spacing), yielding a spatial resolution ranging from 31 km at the Equator to 7 km at 78° latitude. In the third dimension, the model uses 20 non-uniformly spaced depth levels and realistic bottom topography (bathymetry). This is one of the highest resolutions used in any global ocean simulation performed to date. Observed surface winds from the period 1985–1995 and realistic monthly mean heat and salt fluxes are used to force the model.

2 Approach to New Visualization

As POP runs it periodically writes data files representing the progress of the simulation. Although the simulation computes on a 30 minute time-step, these files are written every three days of simulated time. At each three day time-step, one file is written for each variable being computed: salinity, temperature, sea-surface height, flow vector, etc. These dump files represent the results of a simulation.

Historically, we've visualized this sequential collection of data files using video technology. Each file is converted to an image and normalized to some pre-defined color map appropriate for that particular variable. The images are then written to the video device, such as a laser videodisc, one frame at a time.

These video visualizations, while useful for viewing the progress of the simulation, have a serious drawback—they are static and can't be modified without creating a new video. Scientists can't experiment with new color mappings. They can't zoom in on areas of interest. They can't easily specify a section to loop over, nor can they run the video backwards or at variable rates, without scripting the device. Because large simulations are run infrequently, these video animations have long lifespans—their shortcomings become increasingly apparent as time goes on.

Our goal in developing POPTEX was to build a tool for scientists that provides the benefit of video visualizations (putting the results of the simulation into motion) while adding capabilities that enable dynamic, flexible, and interactive exploration of their data. To do this we used the powerful combination of hardware features available on the Laboratory's SGI Origin 2000. Specifically, we sought to exploit the following hardware features:

- **Large Texture Memory.** Four Infinite Reality (iR) graphics pipes [7] are attached to the Origin 2000. Each iR is equipped with 64MB of dedicated texture memory. The images in texture memory can be any of a number of formats (RGBA, intensity, etc.). The texture memory can be allocated to a few large images or a large number of smaller images.

- **Fast Texturing Performance.** Textures can be duplicated on up to four RM7 Raster Manager boards. Each RM7 with its own 64MB of memory contains a full copy of each texture. The parallelism provided by multiple boards enables a near linear speed-up in texturing performance. Equipped with four RM7 boards, the iR can approach textured fill rates of 776M pixels/sec.

- **Texture Lookup Table.** A large texture lookup table (TLUT) provides a level of indirection between between the texture image and the actual colors used to texture the geometry. When using the TLUT, texture images are interpreted as indices into a table of color and alpha values. It is these values that are used to texture the geometry. The TLUT can be loaded much more quickly than texture memory enabling rapid manipulation of texture image mappings.

- **Large Main Memory.** Given our Origin 2000's 16GB of main memory, we can easily pre-load multi-variate data for an entire simulation.

- **Fast Texture Loading.** Once loaded into main memory, the iR supports extremely fast (theoretically 320MB/sec) transfers to its texture memory. This fast transfer allows us to stream time-step images into the texture memory and render them at interactive rates.

We leveraged these hardware features to build POPTEX—a highly interactive tool used to explore the results of the POP ocean simulation.

3 Implementation

A full run of the simulation, representing roughly ten years of elapsed time, generates 1318 time-step dump files. Each dump file contains a copy of the simulation grid with floating point values at each cell location representing variables such as sea-surface height, temperature, and salinity. This 3-D grid is a Mercator projection of the globe, with 1280 cells in longitude and 896 cells in latitude, at each of 20 depths. We preprocess these dump files to extract those variables to be visualized at each depth layer. The dump files are converted to a format more suited to our visualization by normalizing each variable to 8-bit values that will act as indices into the 256-entry texture lookup table. The result of this pre-processing step is a collection of files for each time-step, one file for each variable at every depth layer.

POPTEX itself is written in C++ and uses SGI's version of the OpenGL graphics API. Because OpenGL requires that texture dimensions be a power-of-two, POPTEX splits the 1280×896 input files into two textures that waste the least amount of texture memory—one 1024^2 and another 256×1024. Once these OpenGL textures have been created they can be loaded with images of any size (using `glTexSubImage2DEXT`) as long as they do not exceed the texture's dimensions. In this case, we load them with portions of the input data file—1024×896 into the larger texture and 256×896 into the smaller. An SGI-specific OpenGL extension, `GL_CLAMP_TO_BORDER_SGIS`, is used to ensure that the borders between the two textures are not visible.

These textures are mapped onto a sphere representing the globe. We use a Mercator decomposition of the sphere to more closely represent the simulation grid. The decomposition of the sphere must match the ratio of the two texture sizes so that texture coordinates at the texture's borders coincide with an edge of geometry in the spherical decomposition. Since $1280/256 = 5$ we need only make sure that our decomposition in longitude is divisible by 5. We typically use a decomposition of 40 in latitude and 80 in longitude which produces a visually smooth horizon. From this decomposition we construct OpenGL quadrilateral meshes and compute the texture coordinates at the corner of each quad. Once calculated, the geometry and texture coordinates remain fixed and need not be re-computed. Note that the simulation grid only extends to $\pm 78°$ latitude which accounts for the holes at the poles in this visualization.

Finally, we create the TLUT that will be loaded with specialized colormaps for each variable as we view it. The 8-bit texture values are used as indices into this table and extract the RGBA colors to be used for texture interpolation. This additional level of indirection provides a convenience and a performance improvement—without the TLUT we'd have to reload the entire texture when its colors were changed. This is a much slower operation than reloading a 256 entry TLUT, which can be performed almost instantaneously.

4 Running POPTEX

When a user starts POPTEX it begins loading the time-step data files into memory. This process currently takes about 15 minutes for 1318 time-steps at each of four variables and uses roughly 6GB of main memory. Once all files have been loaded the user can explore the results of the simulation using various facilities provided by the tool.

Any variable may be selected for display by clicking on its corresponding tab. The data for the current time-step is mapped onto the sphere representing the globe. The globe can be rotated and zoomed to any view of interest using a virtual trackball in the main window. Additionally, the current color map for the selected variable is displayed in a panel below the globe. Figure 1 shows sea-surface height selected with low values at the blue end of the spectrum and high values at the red end.

The user can select ranges of values to view, discarding those that they are not interested in, by replacing the contents of the TLUT. The TLUT is modified by manipulating a transfer function which is depicted graphically as an overlay on the current colormap. Iconic handles on the function are dragged to change the function's shape. As the function is being modified the TLUT is continuously reloaded, providing interactive feedback to the user. This can be very useful when certain model features need to be highlighted. For example, Figure 2 shows a view of the salinity of the North Atlantic with all values visible. One of the more interesting water types in the ocean is the salty Mediterranean Sea water. This salty water results from the excess of evaporation compared to precipitation in that region and ends up circulating throughout the world's ocean basins. Figure 3 shows the result of manipulating the transfer function towards the right, or high salinity, side of the colormap so that the modified TLUT highlights areas where Mediterranean Sea water is present.

Although most of the variables we visualize (sea-surface height, temperature, and salinity) are mapped to colors, we have experimented with some alternative mappings. For example, we've used the hillshading technique [5] to display sea-surface height in shaded relief. Figure 4 shows sea-surface height in the western Atlantic using this technique. Visualizations of sea-surface height are of interest in many areas of the world. For example, the strong eddies seen in the Caribbean and Gulf of Mexico can affect the operations of oil drilling platforms.

The main advantage of POPTEX is its animation capability. The collection of sequential data files in main memory can be continuously streamed into texture memory at an observed maximum rate of 72 million texels per second. This results in a maximum frame rate of slightly over 60Hz. At this rate, ten years of simulated time pass in just 21 seconds. More useful than end-to-end animation though, is the ability to choose a period of time and selectively animate over only that range—at any speed, forward of backward, pausing or changing the rate as desired. This technique is particularly useful when studying time-varying phenomena like El Niño. During normal, non-El Niño years, the prevailing winds at the equator tend to push warm water to the west where it collects. When the winds periodically weaken and shift during an El Niño event warm water is allowed to flow back towards the east. Figure 5 shows the warm water pooling in the tropical Pacific while Figure 6 shows the tongue of warm water surging back towards the east only a few months later due to El Niño conditions.

5 Related Work

Nations, et.al. [8] describe a solution to a similar ocean model visualization problem. Unlike us, they specifically eschew any post-processing. Their system runs along with the simulation, visualizing results and helping steer the computation. We reject steering solutions for two reasons. First, the computational scientists are sometimes reluctant to share cycles with the visualization task. Second, simulations often run for weeks to calculate ten years of simulated data while POPTEX can visualize the same ten years of data in just seconds.

The ParVox system at JPL [6] is a general purpose parallel volume rendering system based on the splatting algorithm. They have applied their system to a $640 \times 624 \times 37$ volume of ocean data and achieved rendering rates of one frame per second using 256 processing elements of a Cray T3D.

Vis5D [4] is a software system designed specifically for visualizing numerical climate models like ours. This system performs many of the functions of our system, though not at the rates we achieve by using specific hardware features.

6 Conclusions and Future Work

We have created an interactive tool that allows ocean researchers to interactively visualize their data without creating and viewing video-based animations. In the future we intend to extend the application, adding more features designed specifically to use additional high performance hardware capabilities. Some of these future enhancements are described below:

- **Disk Streaming** – Our first task will be to implement data streaming directly from disk. This will eliminate the tedious process of pre-loading the data, allow the user to select different starting points, and enable the use of data sets that are larger than main memory. In recent experiments we have achieved disk transfer rates of 200MB per second using a striped SCSI file system. We're confident that we can achieve 320MB per second (which matches the theoretical texture load speed of the iR) by striping over more disks and controllers.

- **Volume Rendering** – While visualization of individual depth layers is useful, there are phenomena that may be better understood by viewing all depth layers at once. One example is the Mediterranean outflow. Salty water from the Mediterranean Sea flows through the Strait of Gibraltar, sinking because it's heavier than the less salty water of the Atlantic, and forms a distinct water mass containing numerous eddies at a depth of 1 km. We plan to use the iR's 3-D texture memory for volume rendering [1] on these types of visualizations.

- **Terrain and Bathymetry** – We do not currently display terrain or bathymetry relief. To do so using geometry would quickly surpass even the iR's geometric processing capabilities. We intend to use the aforementioned hillshading technique to produce shaded relief images of global terrain and bathymetry that can then be used as textures and mapped to the sphere along with the simulation data. This should result in a more realistic visualization.

- **Flow Visualization** – POP computes and dumps flow information, in the form of floating point vectors at each grid cell, which we currently do not visualize. We plan to experiment with global flow visualization techniques such as line integral convolution (LIC) [2] to visualize the ocean currents. Sequential LIC images can be pre-computed and streamed onto the globe resulting in a dynamic animation of the flow.

- **Higher Resolution POP Runs** – POP will soon be run at much higher resolution—up to $3500 \times 2500 \times 50$. We plan to apply POPTEX to this high resolution data as soon as it becomes available.

7 Acknowledgments

We would like to thank the other members of the ACL Visualization Team for their support and advice: Jim Ahrens, Pat M^{c}Cormick and Jamie Painter. We would also like to thank additional members of the ocean team, Bob Malone and Rick Smith, for their feedback on early versions of this software. We acknowledge the Advanced Computing Laboratory of Los Alamos National Laboratory, Los Alamos, NM 87545. This work was performed on computing resources located at this facility.

References

[1] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In Arie Kaufman and Wolfgang Krueger, editors, *1994 Symposium on Volume Visualization*, pages 91–98. ACM SIGGRAPH, October 1994. ISBN 0-89791-741-3.

[2] Brian Cabral and Leith (Casey) Leedom. Imaging vector fields using line integral convolution. In James T. Kajiya, editor, *Computer Graphics (SIGGRAPH '93 Proceedings)*, volume 27, pages 263–272, August 1993.

[3] J. K. Dukowicz, R. D. Smith, and R. C. Malone. A reformulation and implementation of the Bryan-Cox-Semtner ocean model on the Connection Machine. *J. Atmos. Ocean. Tech.*, 10:195–208, 1993.

[4] W. Hibbard and D. Santek. The Vis5D system for easy interactive visualization. In *IEEE Visualization '90*, pages 28–35. IEEE, 1990.

[5] B. K. P. Horn. Hillshading and the reflectance map. *Proceedings of the IEEE*, 169(1):14–47, 1981.

[6] P. Peggy Li, Scott Whitman, Roberto Mendoza, and James Tsiao. ParVox—A parallel splatting volume rendering system for distributed visualization. In James Painter, Gordon Stoll, and Kwan-Liu Ma, editors, *IEEE Parallel Rendering Symposium*, pages 7–14, November 1997. ISBN 1-58113-010-4.

[7] John S. Montrym, Daniel R. Baum, David L. Dignam, and Christopher J. Migdal. InfiniteReality: A real-time graphics system. In Turner Whitted, editor, *SIGGRAPH 97 Conference Proceedings*, Annual Conference Series, pages 293–302. ACM SIGGRAPH, Addison Wesley, August 1997. ISBN 0-89791-896-7.

[8] Scott Nations, Robert Moorhead, Kelly Gaither, Steve Aukstakalnis, Rhonda Vickery, Warren C. Couvillion, Jr., Daniel N. Fox, Peter Flynn, Alan Wallcraft, Patrick Hogan, and Ole Martin Smedstad. Interactive visualization of ocean circulation models. In *IEEE Visualization '96*. IEEE, October 1996. ISBN 0-89791-864-9.

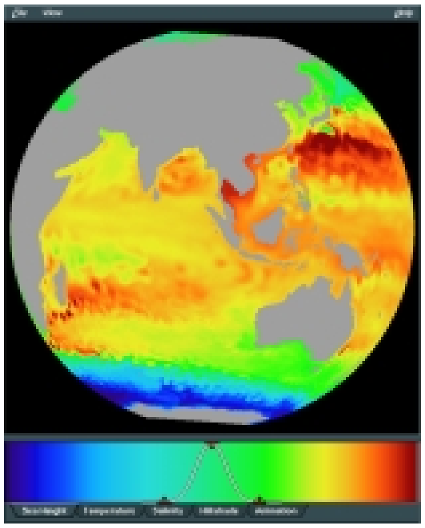

Figure 1: Sea-surface height.

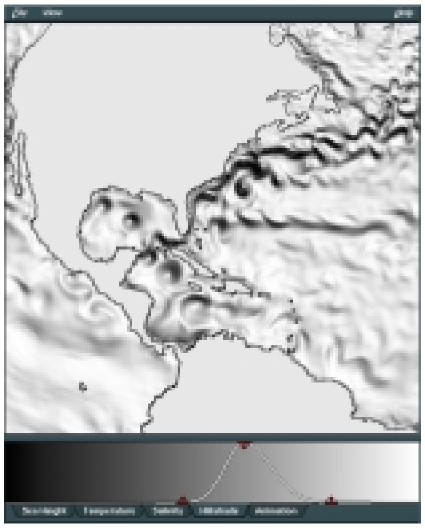

Figure 4: Sea-surface height shown in hillshaded relief.

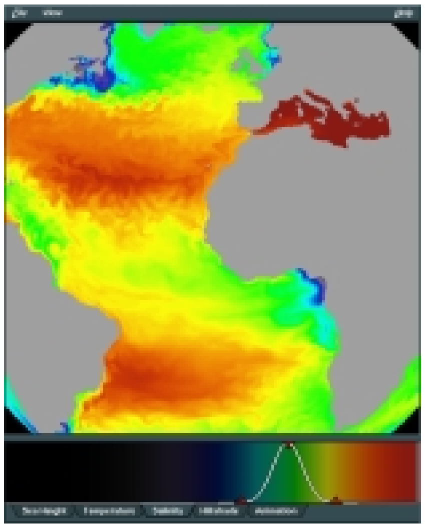

Figure 2: High surface salinity of the Atlantic Ocean.

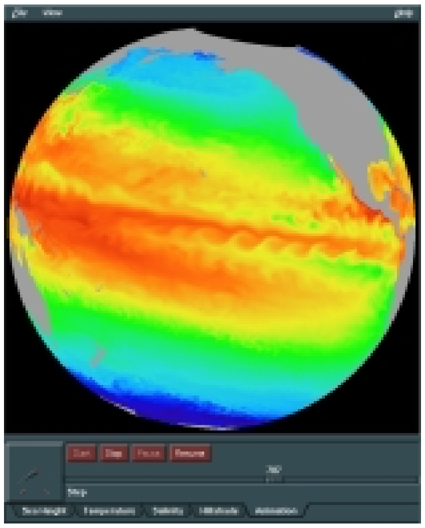

Figure 5: Temperature at surface with no El Niño.

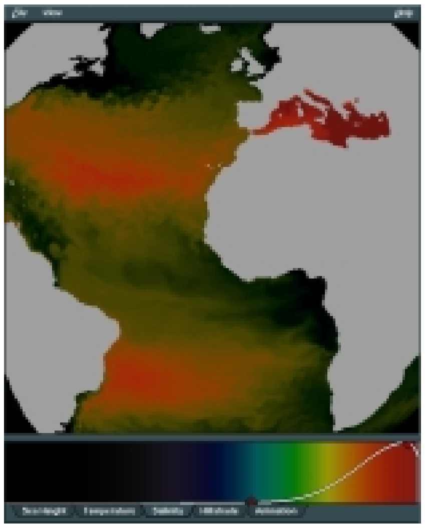

Figure 3: TLUT modified to show areas of high surface salinity.